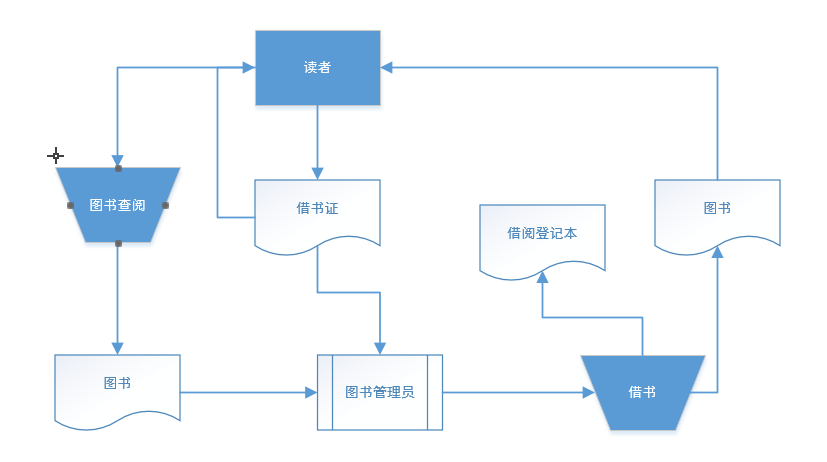
**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于jQuery的xxx系统的分析与设计 | | |
| 课 程 | 软件工程 | | |
| 学 院 | 信息工程学院 | | |
| 专 业 | 软件工程 | 年级 | 2023级 |
| 学生姓名 | 李洋 | 学号 | 012322100005 |

|  |  |  |  |
| --- | --- | --- | --- |
| **评分项** | **评分标准** | **满分** | **得分** |
| 文档整体 | 文档内容详实、规范，美观大方，内容: 可行性分析、需求分析、概要设计、详细设计。 | 20 |  |
| 用例图及规约 | 用例图完整，准确，能够完全体现需求 | 10 |  |
| 顺序图与活动图 | 顺序图能够完全描述用例的设计思路和业务流程，活动图能够描述部分算法的流程。 | 10 |  |
| 类图 | 类图能够完整，准确反映业务的需数据的组织结构 | 10 |  |
| 数据库 | 数据库表的设计来自于类图，合理 | 10 |  |
| 界面设计 | 界面设计美观，清晰，合理，能够完全反映用例图的设计思路 | 10 |  |
| API接口设计 | 为每个界面设计合理的API接口，以便进行前后端分离开发 | 15 |  |
| 内容一致性 | 用例图，类图、数据库、界面相互印证，相互依赖，环环相扣 | 15 |  |
| **得分合计** | | |  |

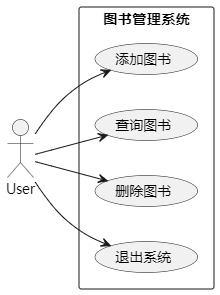
2024 年 12 月 19 日

# 2. 系统总体结构

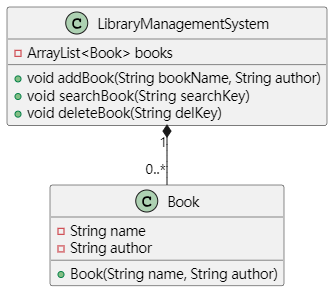




# 3. 用例图设计

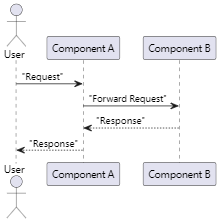


# 4. 类图设计



# 5. 数据库设计

协作图



## book表（图书表）

| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| Books | Arraylist<Book> |  | 否 |  |  | 存储图书对象的列表，LibraryManagementSystem类中定义 |
| Name(Book 类) | String |  | 否 |  |  | 图书的名称在Book  类中定义 |
| author（Book 类） | String |  | 否 |  |  | 图书的名称在Book  类中定义 |
| addBook | viod |  | 否 |  |  | LibraryManagementSystem类中的方法，用于添加图书到books列表 |
| searchBook | viod |  |  |  |  | LibraryManagementSystem类中的方法，用于在books列表中搜索图书 |

​

​

| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| deleteBook | viod | \ |  |  |  | LibraryManagementSystem类中的方法，用于在books列表中删除图书 |
| main | viod |  |  |  |  | LibraryManagementSystem类中的main方法，程序入口，用于处理用户操作选择 |

# 6. 用例及界面详细设计

### 6.1用例规约

| **用例名称** | **修改密码** |
| --- | --- |
| 功能 | 修改用户的密码 |
| 参与者 | 用户 |
| 前置条件 | 必须先登录 |
| 后置条件 | 修改密码以后必须强制登出，再跳转到登录页面 |
| 主事件流 | 1.用户填写新密码（两次输入）  2.用户提交修改信息  3.系统存储修改后密码 |
| 备选事件流 | 1a. 用户两次输入的密码不同     1.系统提示两次输入的密码不相同      2. 用户重新填写并提交。 |

### 6.2业务流程（顺序图）

### 6.3界面设计



### 6.4接口设计

接口1：[setPassword](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E6%8E%A5%E5%8F%A3\\setPassword.md)

用例： [修改密码](file:///D:\CDUcourse_isAnalysis\100ssw\test6\%E7%94%A8%E4%BE%8B\%E4%BF%AE%E6%94%B9%E5%AF%86%E7%A0%81.md)

* 功能： 修改（设置）用户的密码。
* 权限： 用户改自己的密码，必须先登录。
* API请求地址： 接口基本地址/v1/api/setPassword
* 请求方式 ： POST
* 请求实例：

{

"user\_id":1234,

"password":AF#W@#AAAASDF

}

* 请求参数说明:

| **参数名称** | **说明** |
| --- | --- |
| user\_id | 用户的ID号。对应表USERS.USER\_ID的值 |
| password | 用户的密码。不能为空，必须经过加密码,不能是密码的原文。 |

* 返回实例：

{

"status": true,

"info": null,

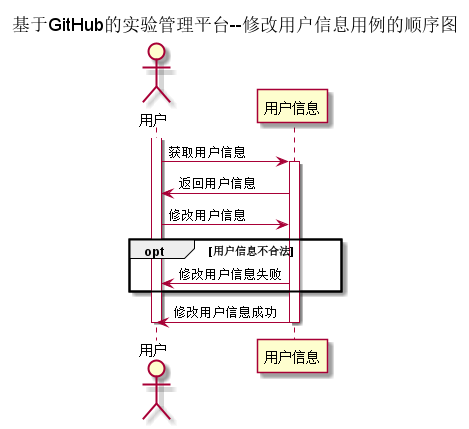
}

## 6.2 [“修改用户信息”用例](file:///D:\CDUcourse_isAnalysis\100ssw\test6\%E7%94%A8%E4%BE%8B\%E4%BF%AE%E6%94%B9%E7%94%A8%E6%88%B7%E4%BF%A1%E6%81%AF.md),[界面](https://ssw383318348.github.io/is_analysis_pages/test6/%E4%BF%AE%E6%94%B9%E7%94%A8%E6%88%B7%E4%BF%A1%E6%81%AF.html)

### 6.2.1用例规约

| **用例名称** | **修改用户信息** |
| --- | --- |
| 功能 | 修改用户的GitHub用户名称 |
| 参与者 | 用户 |
| 前置条件 | 必须先登录，并且查看用户现有的GitHub用户名 |
| 后置条件 |  |
| 主事件流 | 1.用户填写GitHub用户名称  2.用户提交修改信息  3.系统存储修改后的GitHub用户名称 |
| 备选事件流 | 1a. 如果用户输入的GitHub用户名称为空     1.系统清空用户的GitHub用户名称 |

### 6.2.2业务流程（顺序图）



### 6.2.4接口设计

接口1：[setUserInfo](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E6%8E%A5%E5%8F%A3\\setPassword.md)

用例： [修改用户信息](file:///D:\CDUcourse_isAnalysis\100ssw\test6\%E7%94%A8%E4%BE%8B\%E4%BF%AE%E6%94%B9%E5%AF%86%E7%A0%81.md)

* 功能： 修改用户的GitHub用户名。
* 权限： 用户：修改自己的密码，必须先登录。
* API请求地址： 接口基本地址/v1/api/setUserInfo
* 请求方式 ： POST
* 请求实例：

{

"id":"21048329823",

"github\_username":"ABCDE"

}

* 请求参数说明:

| **参数名称** | **说明** |
| --- | --- |
| user\_id | 用户的ID号。对应表USERS.USER\_ID的值 |
| github\_username | 用户GitHub用户名。 |

* 返回实例：

{

"status": true,

"info": null,

}

* 返回参数说明：

| **参数名称** | **说明** |
| --- | --- |
| status | bool类型，true表示正确的返回，false表示有错误 |
| info | 返回结果说明信息 |

## 6.3

### 6.3.4接口设计

图书馆管理系统接口设计

一、系统概述

本图书馆管理系统提供了一组接口，用于实现图书的管理操作，包括添加图书、查询图书、删除图书等功能。这些接口通过命令行与用户进行交互，实现了对图书信息的基本管理。

二、接口设计

（一）用户交互接口

1.  showMenu 接口（显示菜单）

- 功能：向用户展示系统的操作菜单，包括添加图书、查询图书、删除图书和退出系统等选项。

- 输入参数：无。

- 输出结果：在控制台输出以下菜单内容：

- “1. 添加图书”

- “2. 查询图书”

- “3. 删除图书”

- “4. 退出”

- 接口实现示例（伪代码）：

public static void showMenu() {

System.out.println("1. 添加图书");

System.out.println("2. 查询图书");

System.out.println("3. 删除图书");

System.out.println("4. 退出");

}

2.  getUserInput 接口（获取用户输入）

- 功能：从控制台获取用户输入的信息，用于后续操作，如添加图书时获取图书名称和作者，查询和删除图书时获取图书名称等。

- 输入参数：无（通过命令行提示用户输入）。

- 输出结果：返回用户在控制台输入的字符串。

- 接口实现示例（伪代码）：

public static String getUserInput() {

Scanner scanner = new Scanner(System.in);

return scanner.nextLine();

}

（二）图书管理接口

1.  addBook 接口（添加图书）

- 功能：将新图书添加到图书馆管理系统中。

- 输入参数

-  bookName （图书名称， String 类型）：新图书的名称。

-  author （图书作者， String 类型）：新图书的作者。

- 输出结果：在控制台输出“图书添加成功!”表示添加操作完成。

- 接口实现示例（伪代码）：

public static void addBook(String bookName, String author) {

Book book = new Book(bookName, author);

books.add(book);

System.out.println("图书添加成功!");

}

2.  searchBook 接口（查询图书）

- 功能：根据用户输入的图书名称在系统中查找图书并返回相关信息。

- 输入参数

-  searchKey （查询关键字， String 类型）：用户输入的要查询的图书名称。

- 输出结果

- 如果找到图书，在控制台输出“图书名称: [图书名称], 作者: [作者名称]”。

- 如果未找到图书，在控制台输出“未找到该图书”。

- 接口实现示例（伪代码）：

public static void searchBook(String searchKey) {

for (Book book : books) {

if (book.name.equals(searchKey)) {

System.out.println("图书名称: " + book.name + ", 作者: " + book.author);

return;

}

}

System.out.println("未找到该图书");

}

3.  deleteBook 接口（删除图书）

- 功能：从图书馆管理系统中删除指定图书。

- 输入参数

-  delKey （要删除的图书名称， String 类型）：用户输入的要删除的图书名称。

- 输出结果

- 如果删除成功，在控制台输出“图书删除成功”。

- 如果未找到图书，在控制台输出“未找到该图书，无法删除”。

- 接口实现示例（伪代码）：

public static void deleteBook(String delKey) {

for (int i = 0; i < books.size(); i++) {

if (books.get(i).name.equals(delKey)) {

books.remove(i);

System.out.println("图书删除成功");

return;

}

}

System.out.println("未找到该图书，无法删除");

}

（三）系统控制接口

1.  runSystem 接口（运行系统）

- 功能：作为系统的主控制接口，负责启动系统，显示菜单，获取用户选择并调用相应的图书管理接口，循环处理用户操作直到用户选择退出。

- 输入参数：无。

- 输出结果：无（系统根据用户操作在控制台输出相应结果）。

- 接口实现示例（伪代码）：

public static void runSystem() {

while (true) {

showMenu();

String choice = getUserInput();

if (choice.equals("1")) {

System.out.println("请输入图书名称: ");

String bookName = getUserInput();

System.out.println("请输入图书作者: ");

String author = getUserInput();

addBook(bookName, author);

} else if (choice.equals("2")) {

System.out.println("请输入要查询的图书名称: ");

String searchKey = getUserInput();

searchBook(searchKey);

} else if (choice.equals("3")) {

System.out.println("请输入要删除的图书名称: ");

String delKey = getUserInput();

deleteBook(delKey);

} else if (choice.equals("4")) {

break;

} else {

System.out.println("无效的选择，请重新输入");

}

}

}

三、接口调用流程

（一）系统启动

1. 调用 runSystem 接口启动图书馆管理系统。

（二）用户操作

1.  runSystem 接口内部调用 showMenu 接口显示操作菜单。

2. 用户通过控制台输入操作选择， runSystem 接口调用 getUserInput 接口获取用户输入的选择。

3. 根据用户选择：

- 如果选择“1. 添加图书”， runSystem 接口依次调用 getUserInput 接口获取图书名称和作者，然后调用 addBook 接口添加图书。

- 如果选择“2. 查询图书”， runSystem 接口调用 getUserInput 接口获取查询关键字，然后调用 searchBook 接口查询图书。

- 如果选择“3. 删除图书”， runSystem 接口调用 getUserInput 接口获取要删除的图书名称，然后调用 deleteBook 接口删除图书。

- 如果选择“4. 退出”， runSystem 接口结束循环，系统退出。

（三）操作结果输出

1. 图书管理接口（ addBook 、 searchBook 、 deleteBook ）根据操作结果在控制台输出相应信息，如添加成功提示、查询结果或删除结果提示等。

四、接口设计的优点和改进方向

（一）优点

1. 接口功能明确，每个接口专注于特定的操作，易于理解和维护。

2. 通过命令行交互，简单直接，适用于简单的图书管理场景。

（二）改进方向

1. 输入验证增强：当前接口对用户输入的验证较为简单，仅依赖于基本的字符串比较。可以增加更严格的输入验证，例如检查图书名称和作者是否符合特定格式要求，防止非法输入导致系统异常。

2. 错误处理完善：接口在处理异常情况时不够完善，如 Scanner 读取输入可能出现的异常等未进行处理。应添加全面的错误处理机制，提高系统的稳定性和可靠性。

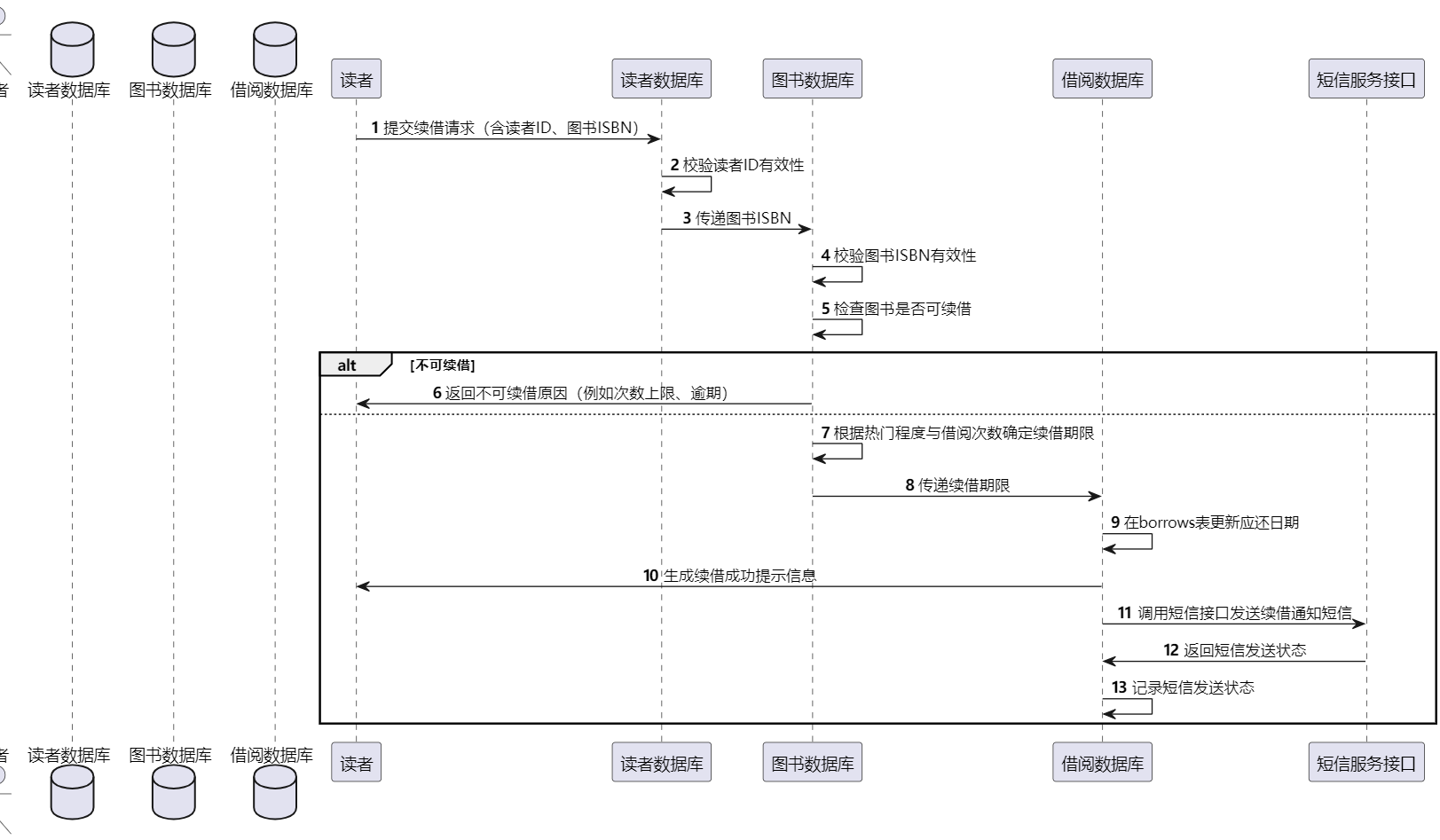
3. 数据持久化支持：系统目前使用 ArrayList 在内存中存储图书数据，数据不持久。可以考虑添加接口用于与数据库或文件存储进行交互，实现数据的持久化管理，以便在系统重启后仍能保留图书信息。

4. 接口扩展性考虑：随着系统功能的扩展，可能需要添加更多图书管理相关操作（如借阅、归还等）或用户管理功能。接口设计应具有一定的扩展性，便于后续功能的添加和修改，例如采用面向接口编程的方式，定义更通用的图书数据操作接口，使系统能够灵活适应不同的存储方式和业务需求。

### 6.4.1用例规约

| **用例名称** | **查看成绩\_老师** |
| --- | --- |
| 功能 | 以表形式的显示出所有用户的信息 |
| 参与者 | 用户 |
| 前置条件 | 用户需要先登录 |
| 后置条件 |  |
| 主事件流 |  |
| 备选事件流 |  |

### 6.4.2业务流程（顺序图）



### 6.6.3界面设计



### 

### 6.7.1用例规约

| **用例名称** | **登录\_用户** |
| --- | --- |
| 功能 | 登录平台 |
| 参与者 | 用户 |
| 前置条件 |  |
| 后置条件 | 登录成功后，跳转到主页 |
| 主事件流 | 1. 访客输入用户名和密码，选择用户类型 2.系统判断用户名，密码，用户类正确，允许登录 3.系统在客户端以Cookie形式存储登录用户信息，保持登录的持久性。 |
| 备选事件流 | 1a. 输入的用户名或者密码为空     1.提示重新输入     2.访客重新提交登录信息  2a.系统判断用户名，密码，用户类不正确，不允许登录     1.提示重新输入     2.访客重新提交登录信息 |